# ACTIVE USER INTERFACES FOR BUILDING DECISION-THEORETIC SYSTEMS

SCOTT M. BROWN

*Crew System Interface Division, Air Force Research Laboratory*
*Wright-Patterson AFB, OH 45433*
*+1 937 255 8883*
*sbrown777@acm.org*

EUGENE SANTOS JR.

*Computer Science and Engineering, University of Connecticut*
*Storrs, CT 06269-3155*
*eugene@cse.uconn.edu*

SHEILA B. BANKS

*Calculated Insight, Orlando, FL*
*sbanks@calculated_insight.com*

**Abstract**

Knowledge elicitation/acquisition continues to be a bottleneck to constructing decision-theoretic systems. Methodologies and techniques for incremental elicitation/acquisition of knowledge especially under uncertainty in support of users' current goals is desirable. This paper presents PESKI, a probabilistic expert system development environment. PESKI provides users with a highly interactive and integrated suite of intelligent knowledge engineering tools for decision-theoretic systems. From knowledge acquisition, data mining, and verification and validation to a distributed inference engine for querying knowledge, PESKI is based on the concept of *active user interfaces* -- actuators to the human-machine interface. PESKI uses a number of techniques to reduce the inherent complexity of developing a cohesive, real-world knowledge-based system. This is accomplished by providing multiple communication modes for human-computer interaction and the use of a knowledge representation endowed with the ability to detect problems with the knowledge acquired and alert the user to these possible problems. We discuss PESKI's use of these intelligent assistants to help users with the acquisition of knowledge especially in the presence of uncertainty.

**Keywords:** Data and Knowledge Engineering/Communication, Human-Agent Interaction

## 1 Introduction

Most everyday decisions involve some level of uncertainty. Expert systems, also known as decision theoretic systems and knowledge-based systems, attempt to capture an expert's knowledge for use by non-experts. Among the advantages to using expert systems are wide distribution, accessibility, and preservation of scarce expertise [1]. One of the greatest disadvantages to expert systems is their

# Report Documentation Page

*Form Approved*
*OMB No. 0704-0188*



| | | |
|---|---|---|
| 1. REPORT DATE<br>**DEC 1999** | 2. REPORT TYPE<br>**N/A** | 3. DATES COVERED<br>**-** |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>**Active User Interfaces For Building Decision-Theoretic Systems** | 5a. CONTRACT NUMBER<br>5b. GRANT NUMBER<br>5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER<br>5e. TASK NUMBER<br>5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>**Air Force Research Laboratory Wright-Patterson AFB, OH 45433-7022** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S)<br>11. SPONSOR/MONITOR'S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT
**Approved for public release, distribution unlimited**

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT<br>**unclassified** | b. ABSTRACT<br>**unclassified** | c. THIS PAGE<br>**unclassified** | **UU** | **10** | |

**Standard Form 298 (Rev. 8-98)**
Prescribed by ANSI Std Z39-18

construction. To aid experts in the arduous task of designing expert systems, a number of expert systems shells exist today. Most of these shells provide a variety of tools to the expert system designers to capture an expert's knowledge, verify and validate that knowledge, and query this knowledge, i.e., perform inference. Some tools provide a graphical means of acquiring knowledge from users. Others incorporate some form of (semi-)automatic verification and validation of the knowledge. However, these tools are designed primarily around the target knowledge representation, that is, the goals of these tools are to satisfy the constraints of the representation and guarantee that the user does so when modifying the knowledge. Furthermore, there is little or no agreed upon methodology for tools interacting with the human user let alone with each other, other than in a haphazard fashion. Thus, none of these systems provide an integrated (human and machine) approach for acquiring knowledge, testing that knowledge via verification and validation, and inference.

A primary goal of our research is to develop a comprehensive software engineering, knowledge engineering, and knowledge elicitation methodology for decision support systems also inclusive of the human element involved. Central to our approach are *active user interfaces* which serve as *actuator*s in the human-machine interface, allowing the user to interact with the computer in a naturalistic/symbiotic manner. Whereas current interface agents are designed around the assistant [14] and/or associate metaphor [11,12], active user interfaces shift the focus from an "agent-centered" system to a "human-centered" system. We emphasize a paradigm of highly interactive collaboration between the human expert and the machine.

Active user interfaces share many ideas from the user modeling and interface agents research fields but also draws heavily from research in the human-computer interaction and artificial intelligence research fields. These interfaces are capable of multi-levels of collaboration and autonomy. The user of an active user interface is fully aware of any actions, whether explicit (authorized consent) or implicit (implied consent), taken by the interface and has a complete, intuitive understanding of such actions.

This paper describes the first application, a probabilistic expert system development environment (PESKI), to benefit from the active user interface approach. (For two other applications that use this approach, see Brown et al. [3].) A suite of intelligent knowledge engineering tools (agents) have been developed and integrated using the active user interfaces paradigm. Simply put, our ultimate goal for PESKI is to guarantee that any and all actions taken by the expert <u>and</u> machine in building a decision support system is done as efficiently as possible, always consistent, and always correct. Given that knowledge engineering is rife with many incremental choices and alternatives at each stage, making the right choices by the human/machine is paramount.

The remainder of this paper is outlined as follows. In the next section we briefly discuss Bayesian Knowledge-Bases (BKBs), a probabilistic knowledge

representation allowing for incremental specification of knowledge. The BKB knowledge representation is capable of determining problems with acquired knowledge. These problems can then be used to actively assist the user in correctly engineering the knowledge-base via the engineering agents. Next, we describe the PESKI system itself and how a user interacts with the various knowledge engineering agents PESKI provides to incrementally acquire knowledge and construct probabilistic expert systems. We conclude with a discussion about future research goals of PESKI and active user interfaces.

## 2 BAYESIAN KNOWLEDGE BASES

To support the design of decision-theoretic systems, we desire to have a knowledge representation that supports modeling uncertainty and is flexible, intuitive, and mathematically sound. A Bayesian knowledge base (BKB) is a probabilistic knowledge representation meeting the preceding qualities [4]. A BKB supports theoretically sound and consistent probabilistic inference — even with incomplete knowledge — with the intuitiveness of "if-then" rule specification. The representation is similar to Bayesian Networks [5]; it is a directed graph capable of representing uncertainty in knowledge via probabilistic relationships between random variables (called components in PESKI). However, Bayesian networks do not allow for incompleteness.

BKBs are built through the combination of instantiation nodes, support nodes, and arcs. An example BKB is shown in Figure 1. *Instantiation nodes*, or *I-nodes* for short, are represented by an oval. An I-node represents one instance of an RV. The arcs represent the relationships between these I-nodes. *Support nodes*, or *S-nodes*, are represented by smaller rectangles or circles. S-nodes are assigned probabilities that are associated with one or more I-nodes. In Figure 1, I-node **Clouds = Heavy** is supported by a single S-node with a probability of 0.1500. I-node **Sidewalk = Wet** is supported by the single I-node **Clouds = Heavy** through an S-node probability of 0.8500. In order for the S-node to be active, the supporting I-node, in this case **Clouds = Heavy**, must be active.

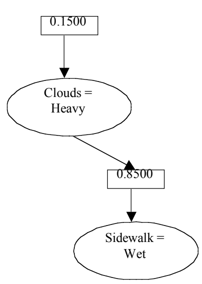


**Figure 1.** Example BKB.

Inherent in the BKB knowledge representation are several consistency constraints endowing the resulting knowledge base the ability to detect problems with the knowledge acquired and which can help alert the user to these possible problems [6]. As a result of these consistency constraints, all knowledge elicited is validated against these constraints. Certain consistency constraint violations can be corrected without user intervention, with an appropriate status message displayed to the user. For others violations, user intervention is required. Users may correct the violation using one of the PESKI intelligent interface agents (e.g., knowledge acquisition, data

mining, verification and validation) discussed next.

## 3 THE PESKI ENVIRONMENT

PESKI (Probabilities, Expert Systems, Knowledge, and Inference) is an integrated probabilistic knowledge-based expert system development environment utilizing Bayesian Knowledge-Bases as its knowledge representation. PESKI provides users with engineering agents for knowledge acquisition [6], verification and validation [7,8], data mining [9], and inferencing [10], each capable of operating in various communication modes to the user.

PESKI consists of four major components:

- <u>Intelligent Interface Agent</u> - translates English questions into inference queries and translates the analyses/inference results back into English as one of its functions; provides for the communication exchange between the user and the system; provide intelligent assistance to the user.
- <u>Inference Engine</u> - contains the intelligent control strategies for controlling the selection and application of various inference engine algorithms (e.g. A*, 0-1 integer linear programming (ILP), genetic algorithms (GAs)) to obtain conclusions to user queries based on knowledge and facts in our knowledge base.
- <u>Explanation & Interpretation</u> - keeps track of the reasoning paths the inference engine used in reaching its conclusions; allows the user to query the system about how and why an answer was derived.
- <u>Knowledge Acquisition & Maintenance</u> - provides the facility for automatically incorporating new or updated expert knowledge into the knowledge base.

Using the active user interfaces paradigm, PESKI is organized into three subsystems (see Figure 2). The four above components serve multiple functions and each PESKI subsystem combines different components together for that subsystem. Let's consider the subsystems as they occur in PESKI in a bit more detail.

- <u>User Interface</u> - composed of the Intelligent Interface and the Explanation & Interpretation components, as well as the interface components for the various engineering agents.
- <u>Knowledge Organization & Validation</u> - consists of the Explanation & Interpretation component along with the human expert and knowledge engineering tools. Organization is accomplished by communicating with the Knowledge Acquisition & Maintenance component, ensuring compliance with the BKB consistency constraints. Validation is similarly accomplished except that we also have feedback from the Reasoning Mechanism through Explanation & Interpretation for debugging purposes.
- <u>Reasoning Mechanism</u> - consists of the Inference Engine and the Knowledge Acquisition & Maintenance components. Our premise behind incorporating the

Knowledge Acquisition & Maintenance component is that we believe some form of reasoning and possibly learning must take place in order for any new knowledge to be merged into our existing knowledge base. Problems such as consistency must be dealt with. Furthermore, we can also achieve a useful degree of information hiding. Under this arrangement, it isn't necessary for any other subsystem outside of the Reasoning Mechanism to concern themselves with our particular choice of knowledge representation. One exception is the Explanation & Interpretation component; however, it only needs to read and interpret knowledge-base information.

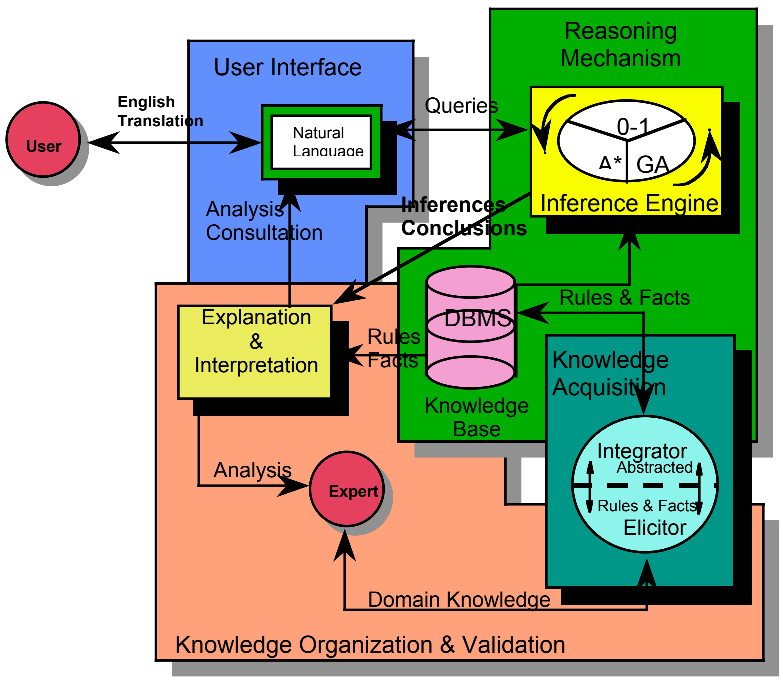



**Figure 2.** PESKI architecture.

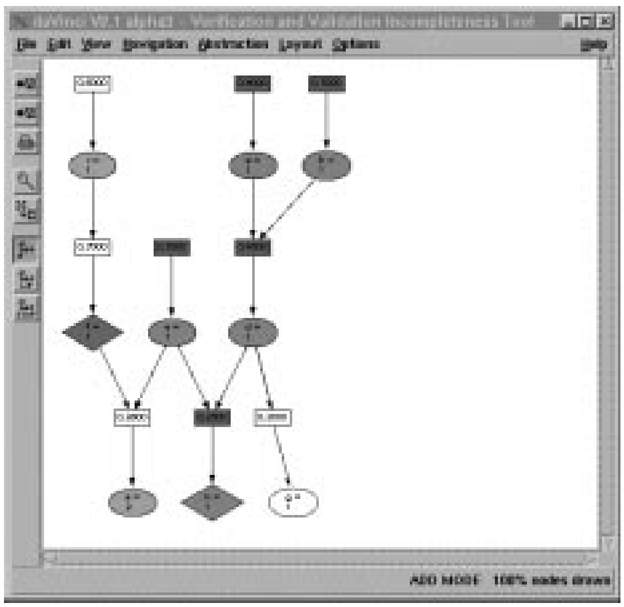
As we can easily see, PESKI provides a complete and integrated suite of knowledge engineering agents for constructing expert systems in nearly any domain. In fact, other alternative architectures can be subsumed by our PESKI architecture [13].

## 4 PESKI's Integrated Engineering Agents Suite

As we have been mentioning thus far, PESKI consists of a suite of engineering agents using the active user interface paradigm. We now briefly describe these agents as they are currently integrated into the PESKI architecture:

- Knowledge Acquisition & Verification - PESKI uses the MACK agent for knowledge acquisition [6]. MACK is designed to automatically and incrementally confirm consistency of the knowledge elicited from the expert and provides assistance by identifying the source of any inconsistency and proactively suggesting corrections. Regular incremental checks preserve both probabilistic validity and logical consistency as knowledge is acquired presumably under the expert's current consideration.
- Validation - PESKI validation is performed using two agents - BVAL [7] and GIT [8]. BVAL validates a knowledge base against its requirements using a test case-based approach. A test case is a set of evidence and expected answers. A knowledge engineer submits a test suite to the BVAL agent and BVAL determines if the given evidence is supported by the answers by submitting a query to the inference engine and comparing the solution with the test case's expected answer. Under certain conditions, the knowledge base is corrected automatically via reinforcement learning of the probabilities. For those test cases that indicate an incompleteness in the knowledge not meeting the test conditions (such as a missing causal relationship between two random variables), the graphical incompleteness tool (GIT) is used to visualize the knowledge base incompleteness for the user and actively provides solutions to correct it. The figure below shows an example of the use of this agent in PESKI. The agent uses data visualization of the BKB and guides the user via color-coded shadings on how to repair the problem.
- Inference Engine - The PESKI inference engine uses a performance metric-based approach to intelligently control a number of possible anytime and anywhere inferencing algorithms (e.g., A*, genetic algorithms). The control is specific to the given knowledge-base and test-case provided by the expert. Results are returned to the user via the Explanation & Interpretation subsystem of PESKI as they become available.
- Data Mining - PESKI uses a goal-directed methodology for data mining for association rules and incorporation of these rules into the knowledge base [9]. Data mining within PESKI can either be a knowledge acquisition or verification and validation process. In the latter case, an expert attempts to correct problems discovered as a result of performing verification and validation. In the former, using empirical and/or legacy data, an expert is able to mine for specific rules relating two or more database attributes (i.e., random variables in the BKB). Additionally, the data mining agent can be used to find new states of an random variable and to elicit the probabilities of a single state.

To recap, PESKI supports incremental knowledge elicitation in a number of ways [6]. During knowledge acquisition, the user is alerted to any inconsistencies in the BKB knowledge representation. For example, if the user attempts to add a rule that creates a cycle in the knowledge base, PESKI will display an error message and MACK will attempt to provide assistance to the user in resolving this.



## 5 Active Intelligent Assistance

As we have described thus far, each agent in PESKI provides active assistance to the expert in the performance of various tasks. Still, these agents are somewhat myopic and focused on collaborating with the user to solve a particular task such as a validation failure for example. Determining which agent to use given a particular situation in PESKI can still be difficult for most users. The use of a particular agent is dependent on a number of variables including the context (e.g., a BKB constraint violation exists) and user preferences for the agents and various communication modes. Determining the correct agent to use at the correct time can be a daunting task.

To aid users in efficiently utilizing the power of the PESKI agent suite offered, we have integrated an intelligent assistant into PESKI applying the active user interface paradigm at this highest level [15,16]. The assistant takes the form of an interface agent, “looking over the shoulder” of the user [14]. The overall goal of the assistant is to offer timely, beneficial assistance to the user as he/she interacts with PESKI. To accomplish this goal, an accurate cognitive model of the user is maintained [17]. The user model captures the goals and needs of the user within the PESKI environment, as well as possible system events that occur, within a probabilistic representation/model of the PESKI environment. Additionally, a user profile is maintained on each user of PESKI so assistance may be custom tailored to individual users. The interface agent determines the how, when, what, and why of offering assistance to the user by inferencing over the user model and utility

functions [3]. The agent acts as a rational decision maker on behalf of the user, using the maximum expected utility principle of decision theory to choose the goal with the maximum expected utility and suggests that goal. The agent is capable of offering assistance for such goals as which agent to use to correct a BKB consistency constraint violation as well as suggesting the user preferred communication mode for a given agent.

We are currently modifying the intelligent agent's architecture to allow it to collaboratively elicit information from the user based on what goals he/she is trying to achieve, his/her preferences, and past actions. To that end, we are adding "deep" domain knowledge of BKBs to the interface agent's user model. Additionally, we are developing an interface agent development environment [19] to assist software developers implement interface agents into other domains.

## 6 Conclusions and Looking to the Future

In this paper, we have discussed the active user interface paradigm of highly collaborative human-machine interaction as applied to knowledge engineering. We have discussed PESKI, an integrated suite of intelligent agents capable of incrementally eliciting additional knowledge from a user as he/she interacts with the system. In conjunction with an overall intelligent interface agent, our goal is to accurately predict the user's intent within PESKI and proactively offer assistance to help the user utilize PESKI in an efficient manner.

The traditional linear lifecycle in the development of decision support systems starting with knowledge elicitation and ending with system validation is a significant part of the engineering bottleneck. As it still typically occurs in practice, each step is wholly completed before proceeding to the next step with only limited step-interaction and expert-interaction. For example, when validation fails, the entirety of the knowledge-base is often returned to the expert with minimal guidance for debugging. With the active user interfaces approach in PESKI emphasizing highly interactive and continuous collaboration between the software agents and the human agents, we redefine the task/lifecycle of knowledge engineering by making it more efficient and more intuitive to the human engineer. For example, the overarching intelligent interface in PESKI learns and adapts to the changing contexts in which the expert is working in. Contexts include obvious items such as input/output preferences to more sophisticated items such as capturing/modeling the subset of the specific domain knowledge that is currently under consideration by the expert. By capturing such contexts, the intelligent interface can better assist the expert at all stages of engineering. For example, the expert can easily "complete" clusters within the domain knowledge by interleaving elicitation and validation on demand. Finally, we believe that the PESKI methodology permits and encourages a distributed approach to knowledge engineering whether through a single expert or potentially multiple experts (see Figure 4). In conclusion, we firmly believe that the active user interfaces paradigm

has tremendous potential for enhancing knowledge engineering such as in terms of increasing the global number of decision support systems in the world to reducing overall time and financial costs to building such systems.

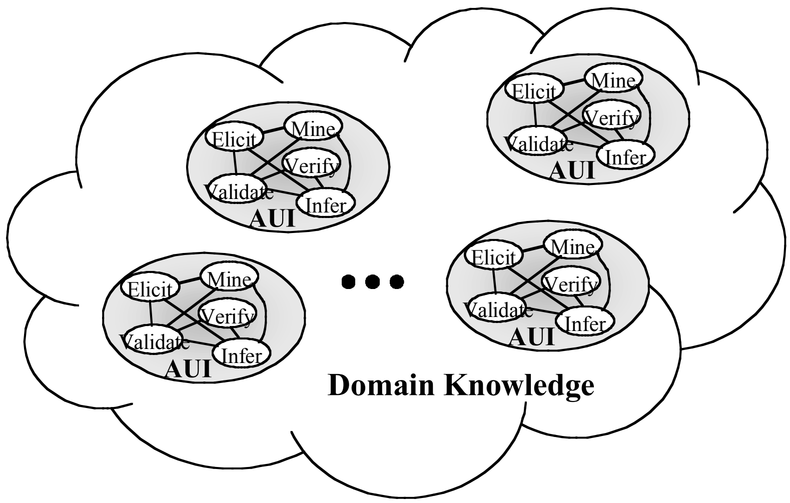



**Figure 4.** Distributed Knowledge Engineering Foci

For more information on PESKI, see http://www.engr.uconn.edu/cse/IDIS.

## 7 ACKNOWLEDGMENTS

PESKI has been the result of years of effort by many students. We thank the entire PESKI team. This research was supported in part by AFOSR Grant Nos. 940006, 9600989, F49620-99-1-0059, and the Office of the Secretary of the Air Force.

## REFERENCES


1. Gonzalez, A.J. and Dankel, D.D., *The Engineering of Knowledge-Based System: Theory and Practice*, Prentice-Hall, Inc., 1993.
2. Banks, S.B., Stytz, M.R., Santos Jr., E., and Brown, S.M., "User Modeling for Military Training: Intelligent Interface Agents," *Proc. of the 19th Interservice/Industry Training Systems and Education Conf.*, 645-653, 1997.
3. Brown, S.M., Santos Jr., E., and Banks, S.B., "Utility Theory-Based User Models for Intelligent Interface Agents", *Proc. of the 12th Biennial Conf. of the Canadian Society for Computational Studies of Intelligence*, 379-393, 1998.
4. Santos Jr., E. and Santos, E. S., "A Framework for Building Knowledge-Bases Under Uncertainty," *Journal of Experimental and Theoretical Artificial Intelligence* **11**, 265-286, 1999.

5. Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988.
6. Santos Jr., E., Banks, D.O., and Banks, S.B., “MACK: A Tool for Acquiring Consistent Knowledge Under Uncertainty,” *Proceedings of the AAAI Workshop on Verification and Validation of Knowledge-Based Systems*, 23-32, 1997.
7. Santos Jr., E., Gleason, H.T., and Banks, S.B., “BVAL: Probabilistic Knowledge-Base Validation,” *Proceedings of the AAAI Workshop on Verification and Validation of Knowledge-Based Systems*, 13-22, 1997.
8. Santos Jr., E., Banks, S. B., Brown, S. M., and Bawcom, D. J., “Identifying and Handling Structural Incompleteness for Validation of Probabilistic Knowledge-Bases,” *Proceedings of the 11th International FLAIRS Conference*, 506-510, 1999.
9. Stein III, D.J., Banks, S.B., Santos Jr., E., and Talbert, M.L., “Utilizing Goal-Directed Data Mining for Incompleteness Repair in Knowledge Bases,” *Proceedings of the 8th Midwest Artificial Intelligence and Cognitive Science Conference*, 82-85, 1997.
10. Shimony, S.E., Domshlak, C., and Santos Jr., E., “Cost-sharing heuristic for Bayesian knowledge-bases,” *Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence*, 421-428, 1997.
11. Banks, S. B. and Lizza, C. S., “Pilot's Associate: A Cooperative, Knowledge-Based System Application,” *IEEE Expert*, 18-29, 1991.
12. Miller, C. A. and Hannen, M. D., “User Acceptance of an Intelligent User Interface: A Rotorcraft Pilot's Associate Example,” *Proceedings of the International Conference on Intelligent User Interfaces*, 109-119, 1999.
13. Buchanan, B. G. and Wilkins, D. C. (Eds.), *Readings in Knowledge Acquisition and Learning: Automating the Construction and Improvement of Expert Systems*, Morgan Kaufmann, 1993.
14. Brown, S.B., Santos Jr., E., Banks, S.B., and Stytz, M.R., “IaDEA: A Development Environment Architecture for Building generic Intelligent User Interface Agents,” *Proceedings of the AAAI Workshop on Software Tools for Developing Agents*, 1998.
15. Harrington, R. A. and Banks, S.B., and Santos Jr., E., “Development of an Intelligent User Interface for a Generic Expert System,” *Proceedings of the 7th Midwest Artificial Intelligence and Cognitive Science Conference*, 1996.
16. Harrington, R. A., Banks, S.B., and Santos Jr., E., “GESIA: Uncertainty-Based Reasoning for a Generic Expert,” *Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence*, 52-55, 1996.
17. Brown, S.M., Santos Jr., E., Banks, S.B., and Oxley, M.E., “Using Explicit Requirements and Metrics for Interface Agent User Model Correction,” *Proceedings of the 2nd International Conference on Autonomous Agents*, 1-7, 1998.
18. Maes, P., “Agents that Reduce Work and Information Overload,” *Communications of the ACM* **37(7)**, 1994, 811-821.